

# Component Documentation

The component is an advanced time selection interface that combines a clean display mode with an interactive scroll-based picker. It features smooth scrolling animations, inline editing capabilities, keyboard navigation, and comprehensive validation for precise time input in 12-hour format.

## How to use

### Basic Usage

A simple time picker with default behavior and display mode.

```
<aava-time-picker
  [size]='lg'
  (timeSelected)="onTimeSelected($event)"
></aava-time-picker>

---

selectedTime = '';

onTimeSelected(time: string) {
  this.selectedTime = time;
  console.log('Selected time:', time);
}
```

### Sizes

Four size options to fit different layout requirements and visual hierarchies.

```
<aava-time-picker size="sm"></aava-time-picker>

<aava-time-picker size="md"></aava-time-picker>

<aava-time-picker size="lg"></aava-time-picker>

<aava-time-picker size="xl"></aava-time-picker>
```

### Available Sizes

- sm (Small) - Compact size for dense interfaces
- md (Medium) - Standard size for most use cases (default)
- lg (Large) - Prominent size for primary actions
- xl (Extra Large) - Extra large size for hero sections and CTAs

### Scroll Mode Interface

Interactive scroll-based time selection with smooth animations and visual feedback.

```

<aava-time-picker></aava-time-picker>

---

selectedTime = '';
scrollEvents: string[] = [];

onTimeSelected(time: string) {
  this.selectedTime = time;
  console.log('Selected time:', time);
}

onScrollEvent(event: any) {
  this.scrollEvents.unshift(
    `Scroll event: ${new Date().toLocaleTimeString()}`
  );
  if (this.scrollEvents.length > 5) {
    this.scrollEvents.pop();
  }
}

```

## Keyboard Navigation

Full keyboard accessibility with arrow keys, number input, and focus management.

```

<aava-time-picker></aava-time-picker>

---

selectedTime = '';
keyboardEvents: string[] = [];

onTimeSelected(time: string) {
  this.selectedTime = time;
  console.log('Selected time:', time);
}

onKeyboardEvent(event: string) {
  this.keyboardEvents.unshift(
    `Keyboard: ${event} - ${new Date().toLocaleTimeString()}`
  );
  if (this.keyboardEvents.length > 5) {
    this.keyboardEvents.pop();
  }
}

```

## Validation and Constraints

Comprehensive validation with proper error handling and boundary enforcement.

```

<aava-time-picker (timeSelected)="onTimeSelected($event)"></aava-time-picker>

<aava-time-picker
  [customStyles]="errorStyles"
  (timeSelected)="onTimeSelected($event)"
>
</aava-time-picker>

---

selectedTime = '';

errorStyles = {
  '--timepicker-background': '#fef2f2',
  '--timepicker-border': '2px solid #ef4444',
  '--timepicker-display-text': '#dc2626',
  '--timepicker-icon-color': '#ef4444',
};

onTimeSelected(time: string) {
  this.selectedTime = time;
  console.log('Selected time:', time);
}

```

## Custom Styling

Customizable appearance with CSS custom properties and theme integration.

```

<aava-time-picker [customStyles]="darkStyles"> </aava-time-picker>

<aava-time-picker [customStyles]="blueStyles"> </aava-time-picker>

<aava-time-picker [customStyles]="greenStyles"> </aava-time-picker>

---

selectedTime = '';

darkStyles = {
  '--timepicker-background': '#374151',
  '--timepicker-border': '1px solid #4b5563',
  '--timepicker-display-text': '#f9fafb',
  '--timepicker-icon-color': '#f9fafb',
  '--timepicker-scroll-background': '#1f2937',
  '--timepicker-time-item-text': '#d1d5db',
  '--timepicker-time-item-selected-text': '#60a5fa',
};

blueStyles = {
  '--timepicker-background': '#dbeafe',
  '--timepicker-border': '1px solid #3b82f6',
  '--timepicker-display-text': '#1e40af',
  '--timepicker-icon-color': '#3b82f6',
  '--timepicker-scroll-background': '#bfdbfe',
  '--timepicker-time-item-text': '#1e3a8a',
  '--timepicker-time-item-selected-text': '#1d4ed8',
};

greenStyles = {
  '--timepicker-background': '#dcfce7',
  '--timepicker-border': '1px solid #22c55e',
  '--timepicker-display-text': '#15803d',
  '--timepicker-icon-color': '#22c55e',
  '--timepicker-scroll-background': '#bbf7d0',
  '--timepicker-time-item-text': '#166534',
  '--timepicker-time-item-selected-text': '#16a34a',
};

onTimeSelected(time: string) {
  this.selectedTime = time;
  console.log('Selected time:', time);
}

```

## Features

### Dual Interface Modes

- Display Mode : Clean, compact time display with click-to-edit functionality
- Scroll Mode : Interactive scroll-based selection with smooth animations
- Seamless Transition : Automatic switching between modes based on user interaction

### Scroll-Based Selection

- Smooth Scrolling : Hardware-accelerated scroll animations with momentum
- Visual Feedback : Clear selection indicators and hover states

- Boundary Enforcement : Prevents scrolling beyond valid time ranges
- Padding System : Intelligent padding for optimal centering and selection

## Inline Editing

- Direct Input : Click any time item to edit directly in place
- Real-time Validation : Immediate validation and formatting
- Keyboard Support : Full keyboard navigation within inline inputs
- Auto-formatting : Automatic formatting and validation on blur

## Advanced Interactions

- Mouse Wheel Support : Scroll through time values with mouse wheel
- Touch Support : Optimized for touch devices and mobile interaction
- Focus Management : Proper focus trapping and restoration
- Event Handling : Comprehensive event management and propagation control

## Accessibility Features

- Keyboard Navigation : Full keyboard accessibility with arrow keys
- Screen Reader Support : Proper ARIA attributes and semantic structure
- Focus Indicators : Clear visual focus indicators
- Reduced Motion : Respects user motion preferences

## API Reference

### Inputs

Property	Type	Description
size	TimePickerSize	Size variant 'sm'   'md'   'lg'   'xl' for the time picker (default: 'md' )

### Outputs

Event	Type	Description
timeSelected	EventEmitter	Emitted when a time value is selected or changed

### Properties

Property	Type	Default	Description
hours	string	"	Current hours value (01-12)

Property	Type	Default	Description
minutes	string	"	Current minutes value (00-59)
period	string	"	Current period value (AM/PM)
displayTime	string	"	Formatted display time string
isFocused	boolean	false	Whether the time picker is in focused state

## ViewChild References

Reference	Type	Description
hoursScroll	ElementRef	Reference to hours scroll container
minutesScroll	ElementRef	Reference to minutes scroll container
periodScroll	ElementRef	Reference to period scroll container

## Internal Properties

Property	Type	Description
showInlineInput	boolean	Whether inline input is currently visible
inlineInputType	'hours'   'minutes'   'period'   null	Type of current inline input
inlineInputValue	string	Current value of inline input
inlineInputPosition	object	Position and dimensions of inline input
clickedItemValue	string	Value of clicked item for inline editing
centeredHour	string	Currently centered hour in scroll view
centeredMinute	string	Currently centered minute in scroll view
centeredPeriod	string	Currently centered period in scroll view

## Readonly Arrays

Property	Type	Description
hoursList	string[]	Array of hours (01-12) with padding
minutesList	string[]	Array of minutes (00-59) with padding
periodList	string[]	Array of periods (AM, PM) with padding

## Methods

### Public Methods

Method	Parameters	Return Type	Description
onDisplayClick()	Event	void	Handles display area click to enter scroll mode
onIconClick()	Event	void	Handles clock icon click to enter scroll mode
onTimeItemClick()	Event, type, value	void	Handles time item click for selection/editing
onScrollEvent()	Event, type	void	Handles scroll events for time selection
centerScrollToValue()	type, value	void	Centers scroll to specific time value
updateDisplayTime()	None	void	Updates the formatted display time
emitTimeSelected()	None	void	Emits the timeSelected event
autoSelectCurrentScrollValues()	None	void	Forces selection of currently visible values

### Private Methods

Method	Parameters	Return Type	Description
generatePaddedList()	string[]	string[]	Generates padded list for smooth scrolling
enforceScrollBoundaries()	type	void	Enforces scroll boundaries for each column
handleWheelEvent()	WheelEvent, type	void	Handles mouse wheel events
updateSelectionForColumn()	type	void	Updates selection for specific column
getFormattedTime()	None	string	Returns formatted time string
onDocumentClick()	MouseEvent	void	Handles document clicks for focus management
onInlineInputChange()	Event	void	Handles inline input value changes
onInlineInputKeyPress()	KeyboardEvent	void	Handles inline input keyboard events
onInlineInputBlur()	None	void	Handles inline input blur events
applyInlineInput()	None	void	Applies inline input value with validation
validateInlineHours()	string	string	Validates and formats hours input
validateInlineMinutes()	string	string	Validates and formats minutes input
validateInlinePeriod()	string	string	Validates and formats period input
closeInlineInput()	None	void	Closes inline input and restores normal view

## CSS Classes

The component provides several CSS classes for styling:

Class Name	Description
.time-picker-container	Main container wrapper
.time-picker-input	Input container with border and styling



Class Name	Description
.time-display	Display mode time text
.time-scroll-container	Scroll mode container
.inline-input-overlay	Inline input overlay container
.inline-scroll-input	Inline input field
.separator	Time separator (:)
.time-scroll-column	Individual scroll column container
.period-column	Period column with smaller width
.scroll-area	Scrollable area container
.time-item	Individual time item
.selected	Selected time item styling
.padding-item	Padding item (invisible)
.hidden-item	Hidden item during inline editing
.icon-wrapper	Clock icon container

## CSS Custom Properties

Property	Description
--timepicker-background	Background color of the time picker
--timepicker-border	Border of the time picker
--timepicker-border-radius	Border radius of the time picker
--timepicker-shadow	Shadow for the time picker container
--timepicker-input-height	Height of the input field
--timepicker-input-min-width	Minimum width of the input field
--timepicker-input-padding-horizontal	Horizontal padding inside input
--timepicker-input-background	Background color for the input
--timepicker-input-border-width	Border width of the input
--timepicker-input-border-color	Border color of the input
--timepicker-input-border-radius-small	Border radius for small input variant
--timepicker-input-padding-small	Padding for small input variant
--timepicker-input-shadow-overlay	Overlay shadow for input field
--timepicker-input-text	Text color for input
--timepicker-display-font-family	Font family for display text

Property	Description
--timepicker-display-text	Text color for display mode
--timepicker-text-to-icon-gap	Gap between display text and icon
--timepicker-scroll-background	Background color for scroll container
--timepicker-scroll-gap-hours-minutes	Gap between hours and minutes in scroll
--timepicker-time-item-height	Height of individual time items
--timepicker-time-item-background	Background color for time items
--timepicker-time-item-padding	Padding for time items
--timepicker-time-item-border-radius	Border radius for time items
--timepicker-time-item-transition	Transition timing for time items
--timepicker-time-item-text	Text color for time items
--timepicker-time-item-hover-background	Background on hover for time items
--timepicker-time-item-hover-text	Text color on hover for time items
--timepicker-time-item-selected-text	Text color for selected time item
--timepicker-time-item-selected-font-weight	Font weight for selected time item
--timepicker-icon-size	Size of the time picker icon
--timepicker-icon-border-radius	Border radius for the icon
--timepicker-icon-padding	Padding inside the icon
--timepicker-icon-color	Icon color
--color-text-primary	Text color used in the separator
--timepicker-size-xsm-height	Height for extra small size variant
--timepicker-size-xsm-min-width	Minimum width for extra small size
--timepicker-size-xsm-padding-horizontal	Horizontal padding for extra small size
--timepicker-border-radius-xs	Border radius for extra small variant
--timepicker-size-xsm-font	Font for extra small size
--timepicker-size-xsm-item-height	Item height for extra small variant
--timepicker-size-xsm-icon-size	Icon size for extra small variant
--timepicker-size-sm-height	Height for small size variant
--timepicker-size-sm-min-width	Minimum width for small size
--timepicker-size-sm-padding-horizontal	Horizontal padding for small size
--timepicker-border-radius-sm	Border radius for small variant
--timepicker-size-sm-font	Font for small size

Property	Description
--timepicker-size-sm-item-height	Item height for small variant
--timepicker-size-sm-icon-size	Icon size for small variant
--timepicker-size-md-height	Height for medium size variant
--timepicker-size-md-min-width	Minimum width for medium size
--timepicker-size-md-padding-horizontal	Horizontal padding for medium size
--timepicker-border-radius-md	Border radius for medium variant
--timepicker-size-md-font	Font for medium size
--timepicker-size-md-item-height	Item height for medium variant
--timepicker-size-md-icon-size	Icon size for medium variant
--timepicker-size-lg-height	Height for large size variant
--timepicker-size-lg-min-width	Minimum width for large size
--timepicker-size-lg-padding-horizontal	Horizontal padding for large size
--timepicker-border-radius-lg	Border radius for large variant
--timepicker-size-lg-font	Font for large size
--timepicker-size-lg-item-height	Item height for large variant
--timepicker-size-lg-icon-size	Icon size for large variant
--timepicker-size-xlg-height	Height for extra large size variant
--timepicker-size-xlg-min-width	Minimum width for extra large size
--timepicker-size-xlg-padding-horizontal	Horizontal padding for extra large size
--timepicker-border-radius-xl	Border radius for extra large variant
--timepicker-size-xlg-font	Font for extra large size
--timepicker-size-xlg-item-height	Item height for extra large variant
--timepicker-size-xlg-icon-size	Icon size for extra large variant

## Best Practices

### User Experience

- Default Values : Set sensible defaults (e.g., current time) for better UX
- Visual Feedback : Ensure clear selection indicators and hover states
- Smooth Transitions : Maintain smooth animations between modes
- Responsive Design : Ensure the component works well on all screen sizes
- Loading States : Handle loading states gracefully

### Performance

- Scroll Optimization : Use hardware acceleration for smooth scrolling
- Event Debouncing : Debounce scroll events to prevent performance issues
- Memory Management : Clean up event listeners and timeouts properly
- Change Detection : Use OnPush strategy for better performance
- Lazy Loading : Load time lists only when needed

## **Accessibility**

- Keyboard Navigation : Ensure all interactions work with keyboard
- Screen Reader Support : Provide proper ARIA labels and descriptions
- Focus Management : Maintain logical focus order and trapping
- Motion Preferences : Respect user's reduced motion preferences
- High Contrast : Ensure visibility in high contrast mode

## **Validation**

- Input Validation : Validate all user inputs thoroughly
- Boundary Enforcement : Prevent invalid time selections
- Error Handling : Provide clear error messages and recovery options
- Format Consistency : Maintain consistent time formatting
- Edge Cases : Handle edge cases like leap seconds, DST changes

## **Content Organization**

- Clear Labels : Use descriptive labels and placeholders
- Logical Grouping : Group related time components logically
- Visual Hierarchy : Use typography and spacing for clear hierarchy
- Consistent Spacing : Maintain consistent spacing throughout
- Responsive Layout : Adapt layout for different screen sizes

## **Accessibility Guidelines**

### **Screen Reader Support**

- ARIA Labels : Provide descriptive labels for all interactive elements
- Live Regions : Use live regions for dynamic content updates
- State Announcements : Announce selection changes and mode switches
- Content Structure : Use semantic HTML structure for better navigation

### **Keyboard Navigation**

- Tab Order : Ensure logical tab order through all interactive elements
- Arrow Keys : Support arrow key navigation within scroll areas
- Enter/Space : Support selection with Enter and Space keys
- Escape : Allow closing or canceling with Escape key
- Focus Indicators : Provide clear visual focus indicators

## **Visual Design**

- High Contrast : Ensure sufficient contrast ratios for all text
- Color Independence : Don't rely solely on color for information
- Text Scaling : Support for text scaling and zoom
- Motion Sensitivity : Provide alternatives for users sensitive to motion

## **Input Validation**

- Clear Messages : Provide clear, actionable error messages
- Real-time Feedback : Give immediate feedback for invalid inputs
- Recovery Options : Provide clear ways to correct errors
- Format Guidance : Show expected input format clearly